Amendments to the Claims:

This listing of claims replaces all prior versions and listings of claims in the application.

Listing of Claims:

1. (Previously Presented) A method for providing access to a resource in a programming environment supporting concurrent tasks, comprising:

(a) providing a latch to a first task, wherein the first task requests the latch to obtain access to the resource;

(b) accessing the resource with the first task;

(c) marking the latch stealable rather than releasing the latch; and

(d) marking the latch unstealable if the first task again requests the latch to obtain access to the resource prior to the latch being requested by a second task seeking access to the resource.

2. (Previously Presented) The method of claim 1, providing (a) further comprising:

(a1) providing to the first task a latch marked stealable and held by a holding task; and

(a2) placing the resource in a consistent state prior to the first task accessing the resource.

3. (Previously Presented) The method of claim 1, further comprising marking a set of flags for a latch stealable, wherein the resource is not placed in a consistent state because the latch is marked stealable, stolen, wherein the resource is placed in a consistent state and then accessed by the second task, or unstealable, wherein the first task again requests the latch to

obtain access to the resource and the latch has not been requested by a second task seeking access to the resource, as requested by the latch.

4. (Cancelled)

5. (Previously Presented) A computer readable medium containing program instructions for providing access to a resource in a programming environment supporting concurrent tasks, the program instructions for:

(a) providing a latch to a first task, wherein the first task requests the latch to obtain access to the resource;

(b) accessing the resource with the first task;

(c) marking the latch stealable rather than releasing the latch; and

(d) marking the latch unstealable if the first task again requests the latch to obtain access to the resource prior to the latch being requested by a second task seeking access to the resource.

6. (Original) The computer readable medium of claim 5, wherein the instructions for providing further comprising instructions for:

(a1) providing to the first task a latch marked stealable and held by a holding task; and

(a2) placing the resource in a consistent state prior to the first task accessing the resource.

7. (Previously Presented) The computer readable medium of claim 6, further instructions for marking a set of flags for a latch stealable, wherein the resource is not placed in a

consistent state because the latch is marked stealable, stolen, wherein the resource is placed in a consistent state and then accessed by the second task, or unstealable, wherein the first task again requests the latch to obtain access to the resource and the latch has not been requested by a second task seeking access to the resource, as requested by the latch.

8. (Cancelled)

9. (Previously Presented) A latch mechanism for a programming environment supporting concurrent tasks, comprising:

means for providing a latch for a resource to a first task, whereby the first task holds the latch;

means for the first task to mark the latch stealable rather than releasing the latch; and

means for the first task to mark the latch unstealable if the first task again requests the latch to obtain access to the resource prior to the latch being requested by a second task seeking access to the latch.

10. (Currently Amended) The latch mechanism of claim 9, wherein the means for providing a latch to the first task further ~~comprising~~ comprises:

means for the first task to be provided, on request, with a latch marked stealable and held by a holding task, and means for placing any resources associated with the latch in a consistent state prior to the first task accessing the resources.

11. (Previously Presented) The latch mechanism of claim 9, further comprising a set of flags for marking a latch stealable, wherein the resource is not placed in a consistent state

because the latch is marked stealable, stolen, wherein the resource is placed in a consistent state and then accessed by the second task, or unstealable, wherein the first task is again provided with the latch and the latch has not been provided to a second task seeking access to the resource.

12. (Cancelled)

13. (Cancelled)

14. (Cancelled)

15. (Cancelled)